On Learning Thresholds of Parities and Unions of Rectangles in Random Walk Models

Sébastien Roch
Department of Statistics
University of California, Berkeley
Berkeley, CA 94720-3860
sroch@stat.berkeley.edu

February 1, 2008

Abstract

In a recent breakthrough, [Bshouty et al., 2005] obtained the first passive-learning algorithm for DNFs under the uniform distribution. They showed that DNFs are learnable in the Random Walk and Noise Sensitivity models. We extend their results in several directions. We first show that thresholds of parities, a natural class encompassing DNFs, cannot be learned efficiently in the Noise Sensitivity model using only statistical queries. In contrast, we show that a cyclic version of the Random Walk model allows to learn efficiently polynomially weighted thresholds of parities. We also extend the algorithm of Bshouty et al. to the case of Unions of Rectangles, a natural generalization of DNFs to $\{0,\ldots,b-1\}^n$.

Keywords: Thresholds of parities, PAC learning, random walk model, statistical queries.

1 Introduction

Learning Boolean formulae in Disjunctive Normal Form (DNF) has been a central problem in the computational learning theory literature since Valiant's seminal paper on PAC learning [25]. In [12], it was shown that DNFs can be learned using membership queries, a form of active learning. Jackson's algorithm, also known as Harmonic Sieve (HS), uses a clever combination of two fundamental techniques in learning, Harmonic Analysis and Boosting. The use of Harmonic Analysis in the study of Boolean functions was introduced in [15]. It was subsequently used as the basis of a learning algorithm for AC^0 circuits in [20]. The Harmonic Analysis used in the HS algorithm is based on a parity-finding algorithm of Goldreich and Levin [10], which was first applied to a learning problem by Kushilevitz and Mansour [19]. Hypothesis boosting, a technique to reduce the classification error of a learning algorithm, was introduced by Schapire [23]. The boosting algorithm used by HS is actually due to Freund [7].

In a recent breakthrough, Bshouty et al. [5] obtained the first passive learning algorithm for DNFs. Their algorithm is based on a modification of HS which focuses on low-degree Fourier coefficients. That variant of HS, called Bounded Sieve (BS), was first obtained in [4]. In [5], BS was used to learn DNFs under the uniform distribution in two natural passive learning models. The first one is the Random Walk model, where examples, instead of being i.i.d., follow a random walk on the Boolean cube (see also [2, 9] for related work). The second model is the closely related Noise Sensitivity model, where this time examples come in pairs, the second instance being a noisy version of the first one. The results of [5] are interesting in that they give a learning algorithm for DNFs in a case where the observer has no control over the examples provided. However the problem of learning DNFs under the uniform distribution when examples are i.i.d. still remains open. It is known that DNFs cannot be learned in the more restrictive Statistical Query model (introduced in [16]) where one can ask only about statistics over random examples [3].

Jackson [12] also showed that HS applies to thresholds of parities (TOP), a class that can express DNFs and decision trees with only polynomial increase in size, and extended his algorithm to the non-Boolean case of unions of rectangles, a generalization of DNFs to $\{0, \ldots, b-1\}^n$ (where $b = O(1)$). Whether those classes of functions can be learned in the Random Walk and Noise Sensitivity models was left open by [5]. Our contribution is threefold. We first show that TOPs cannot be learned in the Noise Sensitivity model using statistical queries (SQs)[1]. As far as we know, this is the first example of a negative result for "second-order" statistical queries, i.e. queries on pairs of examples. This does not rule out the possibility of learning TOPs in the Random Walk model although it provides evidence that the techniques of [5] cannot be easily extended to that

[1] [5] uses only SQs.

case. On the other hand, we show that a simple variant of the Random Walk model where the component updates follow a fixed cycle allows to learn TOPs efficiently. This seems to be the first not-too-contrived passive model in which TOPs are efficiently learnable with respect to the uniform distribution. Actually, one can perform the Harmonic Sieve in this Cyclic Random Walk model, and we also show that this model is strictly weaker than the active setting under a standard cryptographic assumption. Finally we extend the techniques of [4] and [5] to the non-Boolean domain $\{0, \ldots, b-1\}^n$ and use this to learn unions of rectangles in the Noise Sensitivity and Random Walk models. This last result turns out to be rather straightforward once the proper analogues to the Boolean case are found.

In Section 2, we introduce the learning models and give a brief review of Fourier analysis. The negative result for learning TOPs is derived in Section 3. The learning algorithms for TOPs and Unions of Rectangles are presented in Sections 4 and 5 respectively.

2 Preliminaries

We briefly review the learning models we will use and some basic facts about Fourier analysis. For more details see e.g. [18] and [22].

2.1 Learning Models

Let $b \in \mathbb{N}$ be a nonzero constant and let $[b] = \{0, \ldots, b-1\}$. Often we will take $b = 2$. Consider a function $f : [b]^n \to \{1, -1\}$, that we will call the *target* function. Think of f as partitioning $[b]^n$ into positive and negative examples. Denote by U the uniform distribution over $[b]^n$. The goal of the different learning problems we will consider is generally to find for $\varepsilon > 0$ an ε-approximator h to f under the uniform distribution, i.e. a function h such that[2]

$$\mathbb{P}_{x \sim U}[h(x) \neq f(x)] \leq \varepsilon.$$

To achieve this, the learner is given access to limited information which can take different forms.

The **Membership Query** (MQ) model allows to ask for the value of f at any point x of our choosing. The **Uniform Query** (UQ) model on the other hand

[2] For convenience, we will drop the notation $x \sim U$ from probabilities and expectations when it is clear that x is uniform.

works as follows: at any time the learner can ask for an example from f and is provided with a pair $\langle x, f(x)\rangle$ where $x \sim U$; all examples are independent. This type of model is called passive—contrary to the MQ model which is called active—because the learner has no influence over the example provided to him.

In [5], two variants of this model were considered. In the **Random Walk** (RW) model, one is given access to random examples $\langle x, f(x)\rangle$ where the successive values of x follow a random walk on $[b]^n$. Many choices of walks are possible here. We will restrict ourselves to the case where at each step, one component of x, say x_i, is picked uniformly at random and a new value y for x_i is picked uniformly at random over $[b]$ (the first example is uniform over $[b]^n$). A related model is the **Noise Sensitivity** (NS) model. Here a parameter $\rho \in [0,1]$ is fixed and when an example is asked, one gets $\langle x, y, f(x), f(y), S\rangle$ where $x \sim U$ and $y \equiv \mathcal{N}_\rho(x)$ is a noisy version of x defined as follows: for each component of x independently with probability $1-\rho$ a new uniform value over $[b]$ is drawn for this component (we call this operation *updating* and we call $1-\rho$ the *attribute noise rate*), otherwise the component remains the same[3]; S is the set of updated components. We will consider one more variant of these passive models. In the **Cyclic Random Walk** (CRW) model, the successive examples x follow a random walk where at each step, instead of picking a uniformly random component to update, there is a fixed cycle $(i_1, \ldots, i_n)$ running through all of $\{1, \ldots, n\}$ and components are updated in that order (the first example is uniform over $[b]^n$). In all the previous models except MQ, examples are drawn randomly and we therefore allow the learning algorithm to err with probability $1-\delta$ for some δ.

The UQ and NS models also have a **Statistical Query** (SQ) variant. Here, one does not have access to actual examples. Instead in the case of UQ for instance one can choose a polynomial-time computable function $\Gamma : [b]^n \times \{1,-1\} \to \{1,-1\}$ and a tolerance $\tau \in [0,1]$ which is required to be at least inverse polynomially large and the UQ-SQ oracle returns a number γ such that

$$|\mathbb{E}[\Gamma(x, f(x))] - \gamma| \leq \tau.$$

Therefore, the learner can ask only about statistics over random examples. This can be simulated in polynomial time under the UQ model using empirical averaging. But the UQ model is strictly more powerful than UQ-SQ [16]. In the case of NS, the function Γ is allowed to depend on $x, y, f(x), f(y)$. This is called a *second-order* statistical query.

[3] Note that a component is allowed to remain the same even if it is updated.

We will work with three classes of functions. First, we consider Boolean formulae in Disjunctive Normal Form (DNF), in which case $b = 2$. A natural generalization of DNFs to $b > 2$ was given in [12]: for each $1 \leq i \leq n$, choose two values $0 \leq l_i \leq u_i \leq b-1$, and consider the *rectangle*

$$[l, u] = \{x \in [b]^n \ : \ l_i \leq x_i \leq u_i, \ \forall i\}.$$

An instance of UBOX is a union of rectangles. Note that in the Boolean case, a DNF can be seen as a union of subcubes of $[2]^n$. The class of thresholds of parities (TOP) applies only to $b = 2$. A TOP is a function of the form

$$f(x) = \mathrm{sgn}\left(\sum_{m=1}^{M} w_m (-1)^{\sum_i a_i^{(m)} x_i}\right),$$

for M vectors $a^{(m)} \in [2]^n$ and weights $w_m \in \mathbb{Z}$. It is assumed that the weight sum $\sum_{m=1}^{M} |w_m|$ is of size polynomial in n.

We will be interested in learning function classes under the uniform distribution. For any model $\mathcal{M}$, any function class $\mathcal{C}$ and any $\delta, \varepsilon > 0$, we say that $\mathcal{C}$ is (δ, ε)-learnable in $\mathcal{M}$ if there is an algorithm $\mathcal{A}$ such that for any function $f \in \mathcal{C}$ with probability at least $1 - \delta$, $\mathcal{A}$ finds an ε-approximator to f in time polynomial in the description size of f. We say that $\mathcal{C}$ can be weakly learned under $\mathcal{M}$ if there is $\delta > 0$ and ε of the form $\frac{1}{2} - \frac{1}{\mathrm{poly}(n)}$ such that $\mathcal{C}$ can be (δ, ε)-learned in $\mathcal{M}$.

2.2 Fourier Analysis

The complex-valued[4] functions on $[b]^n$ form a linear space where a natural inner product is given by

$$\langle f, g \rangle = \frac{1}{b^n} \sum_x f(x) g^*(x) = \mathbb{E}[f(x) g^*(x)],$$

where * denotes complex conjugation. The set of all generalized parities (parities for short)

$$\chi_a(x) = \omega_b^{\sum_i a_i x_i},$$

where $a \in [b]^n$ and $\omega_b = e^{2\pi i/b}$ form an orthonormal basis and any function can be written as a linear combination

$$f(x) = \sum_{a \in [b]^n} \hat{f}(a) \chi_a^*(x), \tag{1}$$

[4] In the Boolean case, we actually consider only real-valued functions.

where the Fourier coefficient $\hat{f}(a)$ is $\mathbb{E}[f(x)\chi_a(x)]$. A useful result is Parseval's identity

$$\mathbb{E}[|f(x)|^2] = \sum_{a \in [b]^n} |\hat{f}(a)|^2.$$

In learning problems, Fourier-based algorithms usually estimate some of the Fourier coefficients and build an approximation to f in the form of a linear combination as in (1) (and then take the sign or something slightly more complicated in the case $b > 2$). There are two main cases where this technique tends to work. In the "low-degree" case, most of (or at least a non-negligible part of) the Fourier mass is concentrated on low-degree terms, i.e. terms $\hat{f}(a)$ where a has few non-zero components. Then one can estimate all low-degree terms, which can lead to a subexponential algorithm. This is the idea behind the algorithm for learning AC^0 circuits in [20]. In the "sparse" case, most of the mass is concentrated on a few terms. Then one needs to find a way to determine which terms should be estimated. This is the idea behind the algorithm for learning decision trees in [19].

Because one often needs to estimate expectations, e.g. Fourier coefficients, using empirical averages, it is customary at this point to recall Hoeffding's lemma.

Lemma 1 (Hoeffding). *Let X_i be independent random variables all with mean μ such that for all i, $c \leq X_i \leq d$. Then for any $\lambda > 0$,*

$$\mathbb{P}\left[\left|\frac{1}{m}\sum_{i=1}^{m} X_i - \mu\right| \geq \lambda\right] \leq 2e^{-2\lambda^2 m/(d-c)^2}.$$

3 Negative Result for TOPs Learning

For this section, we fix $b = 2$. As demonstrated in [16] and [3], a nice feature of the SQ model is that it allows a complete unconditional characterization of what is learnable under this model. We prove in this section that parities cannot be weakly learned in the Noise Sensitivity model with attribute noise rate at least $\frac{\omega(\log n)}{n}$ (this includes the constant noise rate case used in [5]). This implies in turn that TOPs cannot be weakly learned in this model. Our lower bound on the noise rate is tight for this impossibility result. Indeed, it is easy to see that for an attribute noise rate of $\frac{O(\log n)}{n}$, one can actually learn parities. This follows from the fact that at such a rate, there is a non-negligible probability of witnessing an example $\langle x, y, f(x), f(y), S\rangle$ with exactly one bit flip from x to y, which allows to decide whether the updated variable is contained or not in the parity. One can then repeat for all variables (this can be turned into a statistical query test). In this section, $y = \mathcal{N}_\rho(x)$ with x uniform unless stated otherwise.

We follow a proof of [4]. The main difference is that we need to deal with second-order queries.

Lemma 2. *Any SQ, $\Gamma(x, y, f(x), f(y))$, in the NS model can be replaced by simple expectations, 1st-order queries of the form $\mathbb{E}[g(x)f(x)]$ (where x is uniform), and 2nd-order queries of the form $\mathbb{E}[h(x,y)f(x)f(y)]$ where f is the target function (this actually applies to any second-order SQ model). Moreover, we can assume $|g(x)| \leq 1$ and $|h(x,y)| = 1$ for all $x, y \in [2]^n$.*

Proof. Say we are trying to learn the function f. Because f takes only values -1 and $+1$, we have

$$\begin{aligned}
\mathbb{E}[\Gamma(x,y,f(x),f(y))] &= \mathbb{E}\left[\sum_{i,j=+1,-1} \Gamma(x,y,i,j)\left(\frac{1+if(x)}{2}\right)\left(\frac{1+jf(y)}{2}\right)\right] \\
&= \frac{1}{4}\sum_{i,j=+1,-1}\Big(\mathbb{E}[\Gamma(x,y,i,j)] + i\mathbb{E}_x[f(x)\mathbb{E}_y[\Gamma(x,y,i,j)]] \\
&\qquad + j\mathbb{E}_y[f(y)\mathbb{E}_x[\Gamma(x,y,i,j)]] + ij\mathbb{E}[f(x)f(y)\Gamma(x,y,i,j)]\Big).
\end{aligned}$$

□

Note that the 1st-order queries may not be computable in polynomial time because the averages over x, y are exponential sums (although they might be estimated in polynomial time). But this is not a problem because what we will show is that, no matter what the complexity of the queries is, the *number* of queries has to be superpolynomial. Note also that the simple expectations do not require the oracle (assuming the distribution of x, y is known, as is the case in the NS model). So we ignore them below. Finally, note that in the NS-SQ model, expectations are unchanged if the roles of x and y are reversed.

Following Lemma 2, we can think of a weakly learning algorithm as making a polynomial number of 1st and 2nd-order queries. Denote by s the size of the target function. Say the algorithm $\mathcal{A}$ makes $p(n,s)$ queries with tolerance $1/r(n,s)$ and outputs an $(\frac{1}{2} - \frac{1}{q(n,s)})$-approximator, where the queries are a collection of functions $\{(g_i^{n,s}(x), h_i^{n,s}(x,y))\}_{i=1}^{p(n,s)}$ over $x, y \in [2]^n$ with $|g_i^{n,s}(x)| \leq 1$ and $|h_i^{n,s}(x,y)| = 1$ for all $x, y \in [2]^n$. We now characterize weakly learnable classes in NS-SQ (the characterization actually applies to any second-order SQ model). For this proof, we assume that the confidence parameter $\delta = 0$ (but see the remark after the proof).

Lemma 3. *Let $r'(n,s) = \max\{2r(n,s), q(n,s)\}$. Denote by $\mathcal{C}^{n,s}$ the class of functions in $\mathcal{C}$ restricted to instances of n variables and size at most s. If $\mathcal{C}$ is weakly learnable under NS-SQ (using an algorithm with parameters described above), then there exists a collection $\{V_{n,s}\}_{n,s\geq 1}$ with $V_{n,s}$ of the form*

$$\{(k_i^{n,s}(x), l_i^{n,s}(x,y))\}_{i=1}^{p'(n,s)}$$

with $|k_i^{n,s}(x)| \leq 1$ and $|l_i^{n,s}(x,y)| = 1$ for all $x, y \in [2]^n$, and $p'(n,s) \leq p(n,s)+1$ such that,

$$\forall f \in \mathcal{C}^{n,s},\ \exists i,\ |\mathbb{E}_x[f(x)k_i^{n,s}(x)]| + |\mathbb{E}[f(x)f(y)l_i^{n,s}(x,y)]| \geq \frac{2}{r'(n,s)}. \qquad (2)$$

Proof. We start with $V_{n,s} = \emptyset$. We simply simulate the weak learning algorithm $\mathcal{A}$ with an oracle that returns the value 0 to each query. Every time $\mathcal{A}$ makes a query, we add that query to $V_{n,s}$. At one point $\mathcal{A}$ stops and returns the hypothesis σ. We add $(\sigma, 1)$ to $V_{n,s}$. It is clear that $p'(n,s) \leq p(n,s) + 1$. Assume that (2) is not satisfied. Then there is a function f such that

$$|\mathbb{E}_x[f(x)k_i^{n,s}(x)]| < \frac{2}{r'(n,s)}, \qquad |\mathbb{E}[f(x)f(y)l_i^{n,s}(x,y)]| < \frac{2}{r'(n,s)},$$

for all i. Therefore, in our simulation, the zeros we gave as answers to the queries were valid answers (i.e. within the tolerance $\frac{1}{r(n,s)}$) and therefore because $\mathcal{A}$ returns a weak approximator, it has to be the case that σ is a $(\frac{1}{2} - \frac{1}{q(n,s)})$-approximator. This implies that

$$|\mathbb{E}[f(x)\sigma(x)]| + |\mathbb{E}[f(x)f(y)1]| \geq |\mathbb{E}[f(x)\sigma(x)]| \geq \frac{2}{q(n,s)} \geq \frac{2}{r'(n,s)},$$

a contradiction. □

As noted in [4], because the previous proof does not rely on the uniformity of the learning algorithm and because $\mathrm{BPP} \subseteq \mathrm{P/poly}$, the proof also applies to randomized algorithms.

Theorem 1. *The class of parity functions cannot be weakly learned in NS-SQ with attribute noise rate $\frac{\omega(\log n)}{n}$.*

Proof. Because the size of the function is bounded by a polynomial in n, we drop s from the previous notations. Suppose to the contrary that there is an algorithm $\mathcal{A}$ with parameters as described above that weakly learns parities. By Lemma 3, we have for all $a \in [2]^n$

$$\sum_{i=1}^{p'(n)} \left(\mathbb{E}_x^2[\chi_a(x) k_i^n(x)] + \mathbb{E}^2[\chi_a(x)\chi_a(y) l_i^n(x,y)] \right) \geq \frac{2}{(r'(n))^2}.$$

Taking expectation over uniform $a \in [2]^n$, this is

$$\sum_{i=1}^{p'(n)} \mathbb{E}_a[\mathbb{E}_x^2[\chi_a(x) k_i^n(x)]] + \sum_{i=1}^{p'(n)} \mathbb{E}_a[\mathbb{E}^2[\chi_a(x)\chi_a(y) l_i^n(x,y)]] \geq \frac{2}{(r'(n))^2}.$$

Then either

$$\sum_{i=1}^{p'(n)} \mathbb{E}_a[\mathbb{E}_x^2[\chi_a(x) k_i^n(x)]] \geq \frac{1}{(r'(n))^2}, \tag{3}$$

or

$$\sum_{i=1}^{p'(n)} \mathbb{E}_a[\mathbb{E}^2[\chi_a(x)\chi_a(y) l_i^n(x,y)]] \geq \frac{1}{(r'(n))^2}. \tag{4}$$

In case (3), we get a contradiction by following the same steps as in [4, Theorem 34], which we do not repeat here (their k becomes n and their ρ becomes $\frac{1}{2}$). The attribute noise rate does not play a role in that case. Below, we derive a contradiction out of (4), which follows a similar argument.

From (4) there is an i such that

$$\mathcal{I} = \mathbb{E}_a[\mathbb{E}^2[l_i^n(x,y)\chi_a(x)\chi_a(y)]] \geq \frac{1}{p'(n)(r'(n))^2}. \tag{5}$$

Taking (u,v) to be an independent copy of (x,y), we also have

$$\begin{aligned} \mathcal{I} &= \mathbb{E}_a[\mathbb{E}[l_i^n(x,y)\chi_a(x)\chi_a(y)]\mathbb{E}[l_i^n(u,v)\chi_a(u)\chi_a(v)]] \\ &= \mathbb{E}_{(x,y)}[\mathbb{E}_{(u,v)}[l_i^n(x,y) l_i^n(u,v) \mathbb{E}_a[\chi_a(x \oplus y \oplus u \oplus v)]]], \end{aligned}$$

where $\oplus$ is the parity operator. Denote $\gamma_1 = x \oplus u$ and $\gamma_2 = y \oplus v$. Recall that $|l_i^n(x,y)| = 1$ for all $x, y \in [2]^n$. Then

$$\begin{aligned} |\mathcal{I}| &\leq \mathbb{E}_{\gamma_1,\gamma_2}[|\mathbb{E}_a[\chi_a(\gamma_1 \oplus \gamma_2)]|] \\ &= \mathbb{E}_{\gamma_1,\gamma_2}[|\mathbb{E}_a[\chi_{\gamma_1 \oplus \gamma_2}(a)]|] \\ &= \mathbb{E}_{\gamma_1}[\mathbb{P}_{\gamma_2}[\gamma_1 = \gamma_2]] \\ &= \mathbb{E}_{\gamma_1}\left[\left(\rho^2 + \frac{1}{2}(1-\rho^2)\right)^n\right] \\ &= \left(1 - (1-\rho) + \frac{1}{2}(1-\rho)^2\right)^n. \end{aligned}$$

This last term is the inverse of a superpolynomial if $(1-\rho) = \frac{\omega(\log n)}{n}$, which contradicts (5). □

In the case of constant attribute noise rate, the proof actually implies that even the parities over the first $\omega(\log n)$ variables cannot be weakly learned.

4 Harmonic Sieve in Cyclic Random Walk Model

In this section, we show that HS can be performed efficiently in the CRW model. We also prove that CRW is strictly weaker than MQ under a standard cryptographic assumption.

Theorem 2. *The algorithm HS can be performed in the CRW model with a polynomial increase in time (and an arbitrarily small probability of error).*

As an immediate corollary we get the following.

Corollary 1. *For any $\delta, \varepsilon > 0$, DNFs, TOPs and UBOXs are (δ, ε)-learnable in the CRW model.*

The proof of Theorem 2 follows.

Proof. We only need to check that we can estimate the sums of squares of Fourier coefficients appearing in the Goldreich-Levin algorithm. Without loss of generality, we can rename all components of x so that the components are updated in the order $(n, n-1, \ldots, 1)$. For $1 \leq k \leq n$ and $a \in [b]^k$, let

$$C_{a,k} = \{\hat{f}(ad) \ : \ d \in [b]^{n-k}\},$$

where ad is the concatenation of a and d. Then Jackson [12] showed that it is enough to estimate within inverse polynomial additive tolerance the sum of the squares of terms in $C_{a,k}$ which he also shows to be equal to

$$L_2^2(C_{a,k}) = \sum_{d \in [b]^{n-k}} \hat{f}^2(ad) = \mathbb{E}[\mathrm{Re}(f^*(yx)f(zx)\chi_a(y-z))],$$

where $x \in [b]^{n-k}$, $y \in [b]^k$ and $z \in [b]^k$ are independent uniform, and $y-z$ is taken to be the difference in $\mathbb{Z}_b^k$. In the CRW model, this estimation can be achieved through the following simulation. Make n queries to obtain a uniform instance. Then make $n-k$ queries to update the last $n-k$ bits and get yx and $f(yx)$. Then make k more queries to update the first k bits and get zx and $f(zx)$. It is clear that x, y, z are as required above. From this, compute $\mathrm{Re}(f^*(yx)f(zx)\chi_a(y-z))$. Repeat sufficiently (polynomially) many times and apply Hoeffding's lemma. This takes $2n$ times as many queries as in the MQ model. The rest of the HS algorithm applies without change. Note, in particular, that the boosting part does not require membership queries (see also [4, Theorem 21]). Note also that we didn't assume that f is Boolean above. □

Theorem 3. *If one-way functions exist, the CRW model is strictly weaker than the MQ model.*

Proof. We proceed as in [5, Proposition 2]. If one-way functions exist then there exists a pseudorandom function family $\{f_s : [2]^n \to \{1,-1\}\}_{s \in \{1,-1\}^n}$ [11]. Consider the function g_s which is equal to f_s except on inputs of the form e_i (i.e. the vector with 0's everywhere except on component i where it is 1) where the function is defined as s_i. Then using membership queries, one can learn s from queries to g_s and therefore one can learn g_s. On the other hand, in the CRW model, with probability $1 - 2^{-\Omega(n)}$, one never sees instances e_i's. Therefore if it were possible to learn g_s in this model, this would be essentially equivalent to efficiently learning f_s in the MQ model (by simulation of the conditioned walk) which leads to a contradiction. □

5 Learning Unions of Rectangles

The purpose of this section is to extend the DNF learning algorithm of [5] in the Noise Sensitivity model to the $[b]^n$ setting. The learning algorithm of [5] proceeds in a fashion similar to that of [12] except that it uses *weighted* sums of squared

Fourier coefficients (related to the so-called Bonami-Beckner operator) and considers only $O(\log n)$-degree terms. Therefore the main task in extending this algorithm to UBOXs is to define an appropriate substitute for the Bonami-Beckner operator and show that low-degree terms are also sufficient in this case. The latter was proved by Jackson [12, Corollary 17]. We tackle the former problem in the following theorem.

Theorem 4. *For any $\delta, \varepsilon > 0$, the class of UBOXs is (δ, ε)-learnable in the Noise Sensitivity model, and therefore in the Random Walk model as well.*

Proof. We seek to generalize the weighted sum of squared coefficients used in [5]. A requirement is that it must be possible to estimate the partial sums corresponding to fixing $O(\log n)$ components in the Noise Sensitivity model. A natural choice seems to be

$$(T_\rho f)(x) = \mathbb{E}_{y=\mathcal{N}_\rho(x)}[f(y)],$$

where recall that $\mathcal{N}_\rho(x)$ is a noisy version of x where each component is updated independently with probability $1-\rho$. Here ρ is a fixed constant. Because the operator T_ρ is linear, it suffices to compute its action on the basis functions. Denote by $|S|$ the cardinality of $S \subseteq \{1, \dots, n\}$ and by $|a|$ the number of nonzero components of $a \in [b]^n$. For a vector x and a set S, we note x_S the vector x restricted to components in S, and $0_{S^c}x_S$ signifies the vector which has 0's on components in S^c and is equal to x on components in S. For any $a \in [b]^n$, we have

$$\begin{aligned}
\mathbb{E}_{y=\mathcal{N}_\rho(x)}[\chi_a(y)] &= \frac{1}{b^n} \sum_{z\in[b]^n} \sum_{m=0}^{n} \sum_{S:|S|=m} (1-\rho)^m \rho^{n-m} \chi_a(x + 0_{S^c} z_S) \\
&= \chi_a(x) \sum_{m=0}^{n} \sum_{S:|S|=m} (1-\rho)^m \rho^{n-m} \frac{1}{b^n} \sum_{z\in[b]^n} \chi_{a_S}(z_S) \\
&= \chi_a(x) \sum_{m=0}^{n} \sum_{S:|S|=m} (1-\rho)^m \rho^{n-m} \mathbb{1}\{|a_S| = 0\} \\
&= \chi_a(x) \rho^{|a|} \sum_{m=0}^{n-|a|} \binom{n-|a|}{m} (1-\rho)^m \rho^{n-|a|-m} \\
&= \rho^{|a|} \chi_a(x).
\end{aligned}$$

Therefore,

$$(T_\rho f)(x) = \sum_{a\in[b]^n} \rho^{|a|} \hat{f}(a) \chi_a^*(x).$$

This kind of operator has been used before. See e.g. [14].

We are interested in partial sums of the form

$$\mathcal{T}(I) = \sum_{a:|a_I|=|I|} \rho^{|a|} |\hat{f}(a)|^2,$$

where $I \subseteq \{1, \ldots, n\}$. Indeed, those allow to perform the breadth-first search algorithm in [5, Theorem 7]. Note first that we get a similar upper bound on the weighted Fourier mass of a fixed level of the BFS tree

$$\begin{aligned}
\sum_{I:|I|=j} \mathcal{T}(I) &= \sum_{I:|I|=j} \sum_{a:|a_I|=|I|} \rho^{|a|} |\hat{f}(a)|^2 \\
&= \sum_{a:|a|\geq j} \binom{|a|}{j} \rho^{|a|} |\hat{f}(a)|^2 \\
&\leq \sum_{a:|a|\geq j} |\hat{f}(a)|^2 \sum_{t=j}^{+\infty} \binom{t}{j} \rho^t \\
&\leq \mathbb{E}[|f(x)|^2] \rho^{-1} \left(\frac{\rho}{1-\rho} \right)^{j+1} \\
&\leq \max_x \{|f(x)|^2\} \rho^j (1-\rho)^{-j-1},
\end{aligned}$$

where we have used Parseval's identity. The rest of the proof of [5, Theorem 7] goes without change. The only difference is that now to every $I \subseteq \{1, \ldots, n\}$ corresponds $(b-1)^{|I|}$ vectors $a \in [b]^n$ with $|a_I| = |I|$ and $|a_{I^c}| = 0$. But we can afford to estimate all of them because $|I| = O(\log n)$. Therefore we can find all inversely polynomial coefficients of order $O(\log n)$. Also, we need to check that any UBOX has at least one inversely polynomial coefficient of order $O(\log n)$ and that boosting is possible. This is done in [12, Section 6]. The only point to note is that in the proofs of [12, Fact 14, Corollary 17], one can choose the parity χ_a to have all its components 0 outside the variables included in the $O(\log n)$-rectangle used in the proof (see also [4, Lemma 18]).

It only remains to show that the $\mathcal{T}(I)$'s can be estimated in the Noise Sensitivity model. As in [5], we consider the distribution $\mathcal{D}_\rho^{(I)}$ over pairs $(x, y) \in [b]^n \times [b]^n$ which is $(x, \mathcal{N}_\rho(x))$ conditioned on the event that at least all components in I are updated. This can be simulated in the Noise Sensitivity model by simply picking examples $\langle x, y, f(x), f(y), S \rangle$ until one gets that $I \subseteq S$ (which takes polynomial time if $|I| = O(\log n)$). Then note that

$$\begin{aligned}
\mathcal{T}'(I) &\equiv \mathbb{E}_{\mathcal{D}_\rho^{(I)}}[f(x)f(y)] \\
&= \mathbb{E}_{\mathcal{D}_\rho^{(I)}}\left[\sum_{c,d} \hat{f}(c)\hat{f}(d)\chi_c^*(x)\chi_d^*(y)\right] \\
&= \frac{1}{b^{2n}}\sum_{x,z}\sum_{m=0}^{n-|I|}\sum_{S:|S|=m+|I|}\sum_{c,d}(1-\rho)^m\rho^{n-|I|-m}\hat{f}(c)\hat{f}(d)\chi_c^*(x)\chi_d^*(x+0_{S^c}z_S) \\
&= \frac{1}{b^{2n}}\sum_{x,z}\sum_{m=0}^{n-|I|}\sum_{S:|S|=m+|I|}\sum_{c,d}(1-\rho)^m\rho^{n-|I|-m}\hat{f}(c)\hat{f}(d)\chi_{c+d}^*(x)\chi_{d_S}^*(z_S) \\
&= \sum_{m=0}^{n-|I|}\sum_{S:|S|=m+|I|}\sum_{c,d}(1-\rho)^m\rho^{n-|I|-m}\hat{f}(c)\hat{f}(d)\mathbb{1}\{c+d=0 \bmod b\}\mathbb{1}\{|d_S|=0\} \\
&= \sum_{c}\sum_{m=0}^{n-|I|}\sum_{S:|S|=m+|I|}(1-\rho)^m\rho^{n-|I|-m}|\hat{f}(c)|^2\mathbb{1}\{|c_S|=0\} \\
&= \sum_{c:|c_I|=0}\rho^{|c|}|\hat{f}(c)|^2\sum_{m=0}^{n-|I|-|c|}\binom{n-|I|-|c|}{m}(1-\rho)^m\rho^{n-|I|-m-|c|} \\
&= \sum_{c:|c_I|=0}\rho^{|c|}|\hat{f}(c)|^2,
\end{aligned}$$

where we have used that if f is real and $c+d=0 \mod b$, then

$$\hat{f}(d) = \left(\hat{f}(c)\right)^*.$$

Denote $\mathcal{T}''(I) = \mathcal{T}'(\emptyset) - \mathcal{T}'(I)$. This is

$$\mathcal{T}''(I) = \sum_{c:|c_I|>0}\rho^{|c|}|\hat{f}(c)|^2.$$

We want to estimate $\mathcal{T}(I)$ which consists of a sum over $\{a : |a_I| = |I|\}$. We now know how to estimate the same sum over $\{a : |a_J| > 0\}$ for any J. Noting that $\{a : |a_J| > 0\}$ is made precisely of all $\{a : |a_K| = |K|\}$ with $K \subseteq J$, it is easy to see that $\mathcal{T}(I)$ can be estimated through the $\mathcal{T}''(J)$'s for $J \subseteq I$ by inclusion-exclusion. Since there are only $2^{|I|}$ such J's and $|I| = O(\log n)$, this can be done in polynomial time. The rest of the argument is as in [5, Theorem 11]. □

Ackowledgements

I thank Elchanan Mossel for comments and suggestions. I gratefully acknowledge the financial support of NSERC (Canada) and FQRNT (Quebec, Canada).

References

[1] D. Aldous and U. Vazirani, A Markovian extension of Valiant's learning model, Information and Computation, 117(2):181–186, 1995.

[2] P. Bartlett, P. Fischer, and K.U. Hoffgen, Exploiting random walks for learning, Information and Computation, 176(2):121–135, 2002.

[3] A. Blum, M. Furst, J. Jackson, M. Kearns, Y. Mansour, and S. Rudich, Weakly Learning DNF and Characterizing Statistical Query Learning Using Fourier Analysis, in: Proceedings of the Twenty-sixth Annual ACM Symposium on Theory of Computing, ACM Press, 253–262, 1994.

[4] N.H. Bshouty and V. Feldman, On Using Extended Statistical Queries to Avoid Membership Queries, Journal of Machine Learning Research, 2:359–395, 2002.

[5] N.H. Bshouty, E. Mossel, R. O'Donnell, and R.A. Servedio, Learning DNF from Random Walks, Journal of Computer and System Sciences, 71(3):250–265, 2005.

[6] R. Durrett, Probability: Theory and Examples, Duxbury, 1996.

[7] Y. Freund, Boosting a weak learning algorithm by majority, Information and Computation, 121(2):256–285, 1995.

[8] Y. Freund, M. Kearns, D. Ron, R. Rubinfeld, R. E. Schapire, and L. Sellie, Efficient learning of typical finite automata from random walks, in: Proceedings of the Twenty-fifth Annual ACM Symposium on Theory of Computing, ACM Press, 315–324, 1993.

[9] D. Gamarnik, Extension of the PAC framework to finite and countable Markov chains, in: Proceedings of the Twelfth Annual Conference on Computational Learning Theory, ACM Press, 308–317, 1999.

[10] O. Goldreich and L.A. Levin, A hard-core predicate for all one-way functions, in: Proceedings of the Twenty-first Annual ACM Symposium on Theory of Computing, ACM Press, 25–32, 1989.

[11] J. Hastad, R. Impagliazzo, L. Levin, and M. Luby, A pseudorandom generator from any one-way function, SIAM Journal on Computing, 28(4):1364–1396, 1999.

[12] J. Jackson, An efficient membership-query algorithm for learning DNF with respect to the uniform distribution, Journal of Computer and System Sciences, 55(3):414–440, 1997.

[13] J. Jackson, E. Shamir, and C. Shwartzman, Learning with Queries Corrupted by Classification Noise, Fifth Israel Symposium on Theory of Computing and Systems, 45–53, 1997.

[14] S. Janson, Gaussian Hilbert Spaces, Cambridge University Press, 1997.

[15] J. Kahn, G. Kalai, N. Linial. The influence of variables on boolean functions, in: Proceedings of the 29th Annual Symposium on Foundations of Computer Science, IEEE, 68–80, 1988.

[16] M. Kearns, Efficient noise-tolerant learning from statistical queries, Journal of the ACM, 45(6):983–1006, 1998

[17] M. J. Kearns, Y. Mansour, D. Ron, R. Rubinfeld, R. E. Schapire, and L. Sellie, On the learnability of discrete distributions, in: Proceedings of the Twenty-sixth Annual ACM Symposium on Theory of Computing, ACM Press, 273–282, 1994.

[18] M. J. Kearns and U. V. Vazirani, An Introduction to Computational Learning Theory, MIT Press, 1994.

[19] E. Kushilevitz and Y. Mansour, Learning decision trees using the Fourier spectrum, SIAM Journal on Computing, 22(6):1331–1348, 1993.

[20] N. Linial, Y. Mansour, and N. Nisan, Constant depth circuits, Fourier transforms and learnability, Journal of the ACM, 40(3):607–620, 1993.

[21] Y. Mansour, An $O(n^{\log\log n})$ Learning Algorithm for DNF Under the Uniform Distribution, Journal of Computer and Systems Sciences, 50(3):543-550, 1995.

[22] Y. Mansour, Learning Boolean Functions via the Fourier Transform, in: Theoretical Advances in Neural Computation and Learning, (V.P. Roychodhury and K-Y. Siu and A. Orlitsky, ed.), 391–424, 1994.

[23] R.E. Schapire, The strength of weak learnability, Machine Learning, 5(2):197–227, 1990.

[24] E. Shamir, C. Shwartzman, Learning by Extended Statistical Queries and Its Relation to PAC Learning, Proceedings of the Second European Conference on Computational Learning Theory, Springer, 357–366, 1995.

[25] L. G. Valiant, A theory of the learnable, Communications of the ACM, 27(11), 1134–1142, 1984.